Trabalho de INF1636

13/10/2025

Prof. Ivan Mathias Filho

Instruções para a 2ª iteração

A 2ª iteração tem como objetivo iniciar a implementação de uma interface gráfica por meio da qual se possa jogar Banco Imobiliário.

Nem todas as funcionalidades precisarão ser acionadas, por meio da interface gráfica, nesta iteração. O salvamento e a recuperação de uma partida, por exemplo, deverão ser implementados a partir da 3ª iteração. Sendo assim, o resultado da 2ª iteração será uma versão executável do jogo que pcom algumas limitações.

Interface Gráfica

A interface gráfica do jogo tem de usar componentes Java Swing e Java2D. O tabuleiro, os piões e as cartas têm de ser exibidas, **OBRIGATORIAMENTE**, por meio da **API JAVA 2D**. Já foram disponibilizados arquivos com imagens desses elementos de jogo na página da disciplina no EAD.

É importante ressaltar que **não serão aceitas** soluções baseadas em componentes **Java Swing**, tais como **JPanel**, **JButton**, **JLabel** e etc. As imagens relativas aos elementos de jogo TÊM DE SER EXIBIDAS por meio do método **drawImage()**, definido na classe **Graphics2D**.

As janelas criadas para o jogo devem ter dimensões máximas de 1280 (largura) por 800 (altura) pixels.

Janela Inicial

A primeira janela que será exibida pelo programa permitirá que sejam incluídos os dados dos jogadores (de 3 a 6) de uma nova partida ou a continuação de uma partida cujo estado foi salvo em um arquivo. Feita a escolha, a janela inicial deverá ser fechada e a janela relativa ao tabuleiro deverá ser exibida.

Na inclusão de um jogador, terá de ser informado o seu identificador (uma string com 1 a 8 caracteres alfanuméricos) e a cor do seu pião. O sistema deve impedir que dois jogadores usem piões da mesma cor.

Na 2ª iteração, a implementação da janela inicial deverá contemplar, apenas, a definição do número de jogadores. A continuação de uma partida será abordada a partir da 3ª iteração.

O Tabuleiro

O tabuleiro do jogo **NÃO** pode conter elemento Java Swing algum. Ele deve ser construído **APENAS** com os métodos **fill()**, **draw()** e **drawIamge()**, todos pertencentes à classe **Graphics2D**.

Nenhum componente Java Swing pode ser inserido em um JFrame, ou em um JPanel, com o objetivo de facilitar a construção do tabuleiro do jogo.

As duas únicas exceções são:

- O uso de **JPopupMenu, JMenu** ou **JButton** para o salvamento ou o carregamento de um jogo;
- O uso de **JButton** para disparar a simulação do lançamento dos dados.

As Jogadas

Como regra geral, todas as operações que não precisarem de intervenção do jogador terão de ser feitas automaticamente pelo programa.

A interface gráfica deve indicar, de maneira clara, a cor do jogador da vez. Isso pode ser feito por meio de uma mensagem exibida no próprio tabuleiro ou por meio de cores. A Figura 1 ilustra como é possível indicar o jogador da vez, pintando a área sobre a qual os dados são exibidos com a cor desse (por exemplo, vermelho) jogador.

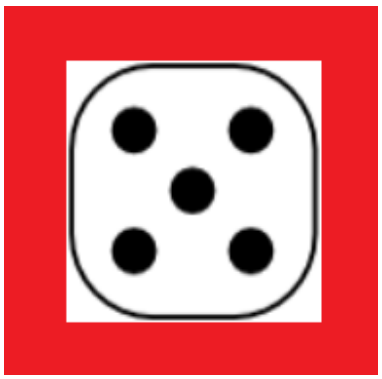

Figura 1 – Indicação do jogador da vez

Evite usar caixas de diálogo (JDilog, JOPtionPane e etc.) para exibir simples mensagens, pois elas exigem a intervenção do jogador, o que atrapalha o andamento das jogadas.

O resultado da simulação do lançamento dos dados deve ser exibido por meio de figuras que representem os números obtidos no lançamento. Todas as seis figuras encontram-se publicadas na página da disciplina no EAD.

Observação: para testar o comportamento do jogo em situações específicas é desejável que os valores dos dados possam ser definidos pelo testador, em vez de serem definidos por meio de uma função de randomização. Sendo assim, já na 2ª iteração, deve ser implementada uma forma de definição dos valores dos dados pelo testador. Sugestão: use dois combo boxes que contenham os inteiros de 1 a 6.

Deslocamento dos Piões

Sugestão: crie 6 pistas imaginárias que cubram todo o perímetro do tabuleiro. Dessa forma, não será necessário criar um algoritmo para posicionar os piões na hipótese de dois ou mais deles estarem ocupando uma mesma casa.

Deck de Cartas de Sorte/Revés

O deck de cartas de sorte/revés não precisa ser exibido junto ao tabuleiro, para não ocupar espaço no painel. Entretanto, a carta que acabou de ser recebida por um jogador deve ser exibida para todos. Você pode exibi-la em alguma área do painel do tabuleiro ou abrir uma caixa de diálogo para tal.

As imagens dos textos das cartas farão parte do componente View da arquitetura MVC. O componente Model deverá representar o significado de cada carta. Além disso, é desejável que cada carta presente no Model possua um identificador que permita à View recuperar a imagem correspondente em tempo constante (por exemplo, uma chave em um HashMap).

Informações Sobre o Estado do Jogo

Durante a realização de uma partida várias informações devem estar disponíveis para consulta.

Propriedades

Quando um pião for posicionado em uma propriedade, como resultado de seu deslocamento, informações sobre essa propriedade devem ser exibidas. São elas:

- A carta relativa à propriedade;
- Caso ela tenha proprietário, a cor dele;
- O preço de compra da propriedade;
- Caso a propriedade seja um terreno, o número de casas e hotéis construídos.

Propriedades e Finanças do Jogadores

Os dados sobre as propriedades e os recursos financeiros do jogador da vez devem ser exibidos (e atualizados) enquanto ele estiver jogando. Você pode exibir essas informações no próprio painel do tabuleiro ou abrir uma janela para tal. Caso opte por exibir os dados do jogador da vez no próprio painel do tabuleiro, sugiro que o montante de dinheiro que ele possui seja exibido como um simples texto, enquanto que a relação de propriedades seja exibida em uma combo box. Caso se deseje examinar os dados de uma dessas propriedades bastará selecioná-la na combo box e exibir os seus dados, no próprio painel do tabuleiro ou em outra janela.

Operações do Jogador da Vez

Durante a realização de uma jogada, o jogador da vez poderá construir uma casa ou um hotel, além de vender propriedades para o banco. Para tal, será necessário criar diálogos específicos para que essas operações sejam realizadas. Além disso, será necessário usar push buttons (JButton) ou menus para ativar essas operações. Você está livre para implementar esses diálogos da maneira que achar melhor. O uso de push buttons e menus é apenas uma sugestão.

Quando uma operação de débito (ou de crédito) for realizada, o programa deve informar o montante debitado (ou creditado) do jogador da vez, quem recebeu o crédito (ou débito) e o saldo financeiro de quem recebeu o crédito (ou débito). Nada precisa ser acrescentado sobre o jogador da vez, visto que já foi mencionado que todos os dados sobre ele devem ser exibidos e atualizados durante a sua jogada.

Devido a um jogador poder realizar várias operações durante a sua jogada, TALVEZ seja necessário utilizar um push button (ou outro componente que você considerar adequado) para que o jogador da vez possa encerrar a sua jogada e passar a vez para o próximo jogador.

Funcionalidades

As seguintes funcionalidades terão de ser implementadas na 2ª iteração por meio de uma interface gráfica:

- Exibição da janela inicial;
- Definição dos jogadores de uma partida;
- Exibição do tabuleiro do jogo;
- Definição da ordem dos jogadores por meio de sorteio;
- Lançamento dos dados e movimentação dos piões;
- Exibição das cartas contendos os dados das propriedades.

Design e Implementação

Na avaliação do trabalho será levada em consideração a aplicação correta das técnicas de design e programação vistas durante o curso. Isso inclui a observação dos critérios de acoplamento e coesão, **a organização do aplicativo em pacotes** e a utilização OBRIGATÓRIA dos seguintes Design Patterns:

- Observer
- Façade
- Singleton

Nenhuma referência a elementos gráficos, elementos da View, poderá ser feita diretamente a partir de métodos localizados no Model. A atualização das janelas, de modo a refletir o resultado da última jogada, terá de ser feita, OBRIGATORIAMENTE, por meio do padrão Observer.

O Controller será o responsável pela abertura de **JOptionPanes**, em caso de exibição de mensagens relevantes ao andamento de uma rodada, e de **JFileChoosers**, para que o estado de um jogo seja salvo.

As instruções desta iteração têm por objetivo apresentar uma visão global da arquitetura que será implementada. Entretanto, não há compromisso algum de que elas sejam plenamente atendidas na 2ª iteração.